UNIT PEMODENAN TADBIRAN DAN PERANCANGAN PENGURUSAN MALAYSIA

Jurnal *"On Boarding"* Sistem eRating ke Platform Openshift
October 2018

1.0 Pengenalan

Dokumen ini adalah langkah implimentasi UAT sistem eRating pada platform Red hat Openshift Container semasa penglibatan pasukan projek eRating.Ia mengandungi perincian keputusan aktiviti UAT yang telah dilaksanakan oleh pasukan projek semasa aktiviti *onboarding*.

2.0 Development Instance Pipeline

2.1 eRating Dev Pipeline

```
try {
    timeout(time: 20, unit: 'MINUTES') {
    def appName="erating"
    def projectDev="erating-dev"
    def projectCicd="erating-cicd"
    def gitRepo="https://github.com/shahrizanrajak/erating_onboard.git"

    properties([
        pipelineTriggers([
            pollSCM('5 * * * *')
        ])
    ])

node () {
        stage ("Prepare Build"){
            git branch: "master", url: gitRepo , poll: true, changelog: true
        }
        stage("Deploy Dev") {
            sh "oc project ${projectDev}"
            sh "oc delete dc,svc,route -l app=${appName} -n ${projectDev} || true"
            sh "oc new-app php-70-redis~${gitRepo} --name=${appName}  || true"
            sh "oc rollout cancel dc/${appName} -n ${projectDev} || true"
            sh "oc rollout pause dc/${appName} -n ${projectDev} || true"
            sh "oc set volume dc/${appName} -n ${projectDev} --add --name=database-config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/database.php --sub-path=database.php || true"
            sh "oc set volume dc/${appName} -n ${projectDev} --add --name=config -t configmap --
configmap-name=erating-cm --mount-path=/opt/app-root/src/application/config/config.php --sub-
path=config.php || true"
            sh "oc start-build ${appName} --wait"
            sh "oc rollout resume dc/${appName} -n ${projectDev}"
            sh "oc rollout status dc/${appName} -w -n ${projectDev}"
            sh "oc expose svc/${appName} -n ${projectDev} "
        }

}
}
}catch (err) {
    echo "in catch block"
    echo "Caught: ${err}"
    currentBuild.result = 'FAILURE'
    throw err
}
```

Proses yang berlaku di *pipeline* diatas adalah:

1. Poll SCM akan aktif untuk memeriksa SCM (Git) setiap 5 minit.

Jika terdapat sebarang perubahan akan mengaktifkan *pipeline*

secara automatic.

2. Hapus *Service, Deployment* dan *route* lama

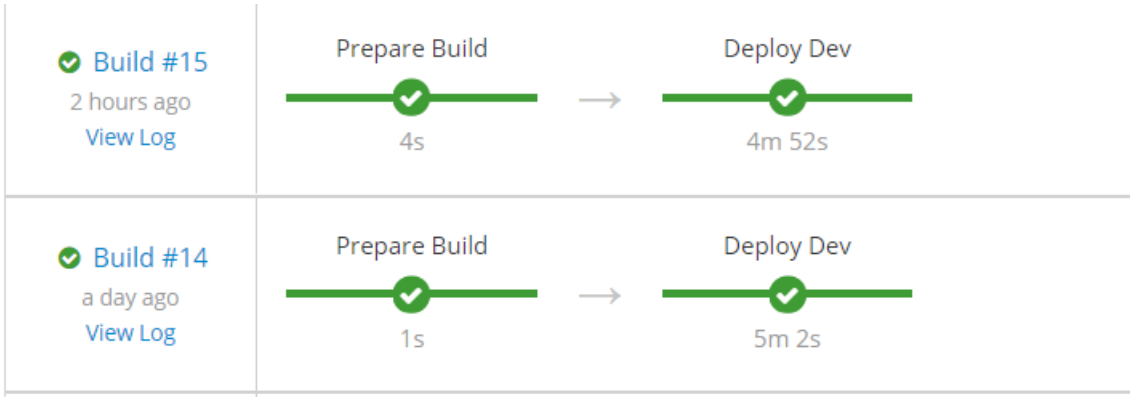
3. Cipta aplikasi baru untuk eRating berdasarkan kod dari Git

4. Henti sementara pelancaran dan setkan fail konfigurasi menggunakan *volume* dari peta konfigurasi

5. Teruskan pelancarran dan tunggu sehingga tamat

6. *Expose* aplikasi tersebut.

2.2 Hasil perlaksanaan *Pipeline* eRating Dev (Pembangunan)



3.0 SIT Instance *Pipeline*

3.1 eRating SIT Pipeline

```
try {
    timeout(time: 20, unit: 'MINUTES') {
    def appName="erating"
    def projectDev="erating-dev"
    def projectSit="erating-sit"
    def databaseSvc="mysql"
    def gitRepo="github.com/shahrizanrajak/erating_onboard.git"
    def version

    node () {

      stage("Promote SIT") {
          timeout(time:4, unit:'HOURS') {
              env.VERSION = input message: 'Promote to SIT?', ok: 'Promote!',
                          parameters: [string(name: 'VERSION', defaultValue: '', description:
'VERSION')]
```

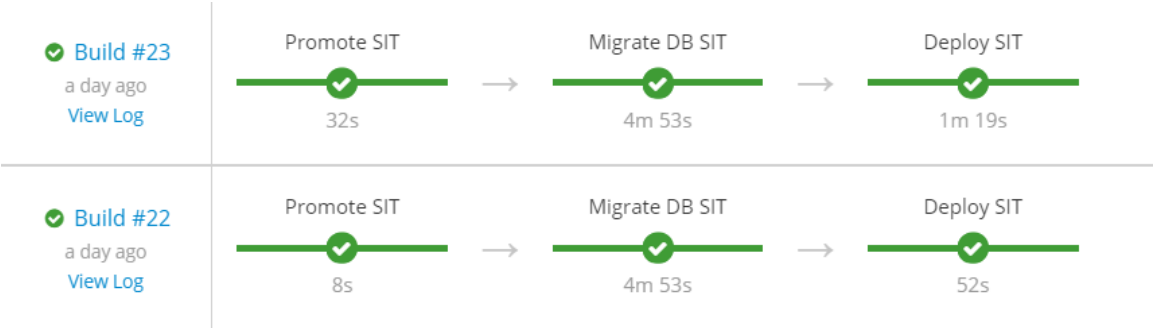
```
            }
            sh "oc create imagestream ${projectSit} || true"
            sh "oc tag ${projectDev}/${appName}:latest ${projectDev}/${appName}:${env.VERSION}"
            sh "oc tag ${projectDev}/${appName}:${env.VERSION} ${projectSit}/${appName}:latest"
            sh "oc tag ${projectSit}/${appName}:latest ${projectSit}/${appName}:sit-${env.VERSION}"
        }
        stage("Migrate DB SIT") {
            sh "oc get pod -o jsonpath='{.items[?(@.status.phase == \"Running\")].metadata.name}' -l
deploymentconfig=${databaseSvc} -n ${projectSit} > mysqlpod"
            mysqlpod = readFile('mysqlpod').trim()
            sh "oc exec -n ${projectSit} ${mysqlpod} -- bash -c 'mysqldump -uroot -
pBPiKmuj4vVCwMSKm -hmysql.erating-dev.svc erating|mysql -uroot -p7C7QVx5kAf66PlDy -
hmysql erating'"
        }
        stage("Deploy SIT") {
            sh "oc export cm -n ${projectDev} | oc create -n ${projectSit} -f - || true"
            sh "oc delete dc,svc,route -l app=${appName} -n ${projectSit}"
            sh "oc new-app ${appName}:sit-${env.VERSION} -n ${projectSit}"
            sh "oc rollout cancel dc/${appName} -n ${projectSit} || true"
            sh "oc rollout pause dc/${appName} -n ${projectSit}"
            sh "oc set volume dc/${appName} -n ${projectSit} --add --name=database-config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/database.php --sub-path=database.php"
            sh "oc set volume dc/${appName} -n ${projectSit} --add --name=config -t configmap --
configmap-name=erating-cm --mount-path=/opt/app-root/src/application/config/config.php --sub-
path=config.php"
            sh "oc rollout resume dc/${appName} -n ${projectSit}"
            sh "oc rollout status dc/${appName} -w -n ${projectSit}"
            sh "oc expose svc/${appName} -n ${projectSit}"
        }

    }
}
} catch (err) {
    echo "in catch block"
    echo "Caught: ${err}"
    currentBuild.result = 'FAILURE'
    throw err
}
```

Proses yang berlaku di *pipeline* diatas adalah:

1. Maklum kepada pengguna untuk memasukkan versi.
2. Salin imej dari Pembangunan kepada SIT dengan tanda terbaharu
3. Tanda imej dalam SIT Instance pada Tanda versi
4. DB (pangkalan data) dimigrasi dari pembangunan kepada SIT
5. Exsport fail peta konfigurasi dari pembangunan kepada SIT
6. Padam *Service, Deployment* dan *Route* lama.

7. Cipta app baharu berdasarkan imej yang telah disalin.
8. Hentikan sementara *Rollout* dan setkan file konfigurasi berdasarkan *volume* menggunakan peta konfigurasi.
9. Sambung Rollout dan tunggu hingga tamat.
10. *Expose* aplikasi.

3.2 Hasil perlaksanaan eRating SIT *Pipeline*



1. Aplikasi *Backend* dihentikan sementara dan setkan semula pembolehubah persekitaran supaya boleh dibaca didalam fail *config* aplikasi
2. *Rollout* disambut semua sehingga tamat.
3. *Expose* aplikasi

4.0 UAT Instance Pipeline

4.1 eRating UAT Pipeline

```
try {
    timeout(time: 20, unit: 'MINUTES') {
    def appName="erating"
    def projectUat="erating-uat"
    def projectSit="erating-sit"
    def databaseSvc="mysql"
    def gitRepo="github.com/shahrizanrajak/erating_onboard.git"
    def version

    node () {
```

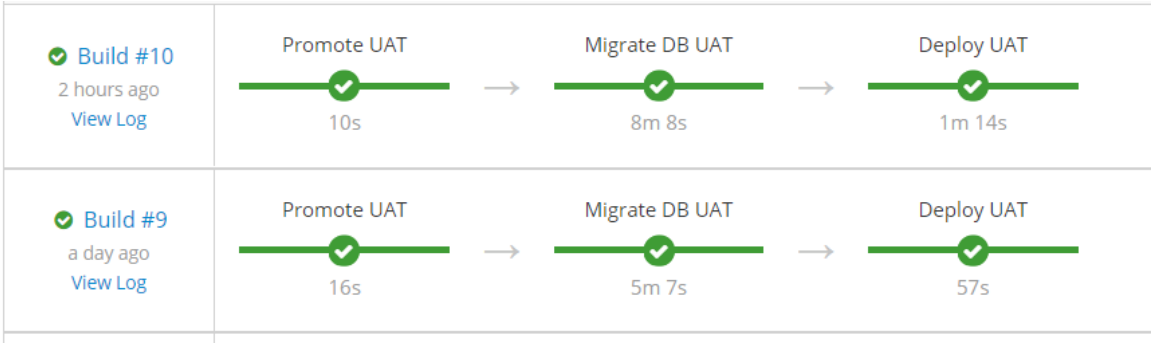
```
    stage("Promote UAT") {
        timeout(time:4, unit:'HOURS') {
            env.VERSION = input message: 'Promote to UAT?', ok: 'Promote!',
                    parameters: [string(name: 'VERSION', defaultValue: '', description:
'VERSION')]
        }
        sh "oc create imagestream ${projectUat} || true"
        sh "oc tag ${projectSit}/${appName}:sit-${env.VERSION} ${projectUat}/${appName}:Uat-
${env.VERSION}"

      }
      stage("Migrate DB UAT") {
        sh "oc get pod -o jsonpath='{.items[?(@.status.phase == \"Running\")].metadata.name}' -l
deploymentconfig=${databaseSvc} -n ${projectUat} > mysqlpod"
        mysqlpod = readFile('mysqlpod').trim()
        sh "oc exec -n ${projectUat} ${mysqlpod} -- bash -c 'mysqldump -uroot -
p7C7QVx5kAf66PlDy -hmysql.erating-sit.svc erating|mysql -uroot -p4DjW5OXynQvIHC4D -
hmysql erating'"
      }
      stage("Deploy UAT") {
        // clean up. keep the image stream
        sh "oc export cm -n ${projectSit} | oc create -n ${projectUat} -f - || true"
        sh "oc delete dc,svc,route -l app=${appName} -n ${projectUat}"
        sh "oc new-app ${appName}:uat-${env.VERSION} -n ${projectUat}"
        sh "oc rollout cancel dc/${appName} -n ${projectUat} || true"
        sh "oc rollout pause dc/${appName} -n ${projectUat}"
        sh "oc set volume dc/${appName} -n ${projectUat} --add --name=database-config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/database.php --sub-path=database.php"
        sh "oc set volume dc/${appName} -n ${projectUat} --add --name=config -t configmap --
configmap-name=erating-cm --mount-path=/opt/app-root/src/application/config/config.php --sub-
path=config.php"
        sh "oc rollout resume dc/${appName} -n ${projectUat}"
        sh "oc rollout status dc/${appName} -w -n ${projectUat}"
        sh "oc expose svc/${appName} -n ${projectUat}"
      }

   }
}
} catch (err) {
    echo "in catch block"
    echo "Caught: ${err}"
    currentBuild.result = 'FAILURE'
    throw err
}
```

Proses pipeline diatas adalah:

1. Minta pengguna memasukkan versi aplikasi
2. Salin imej dari Instance SIT ke Instance UAT berdasarkan versi yang telah dimasukkan.
3. Migrate DB dari SIT ke UAT.

4. Export fail config map dari SIT ke UAT.
5. Hapus servis lama, *Deployment*, dan *route*.
6. Cipta aplikasi baru berdasarkan imej yang telah disalin.
7. Hentikan sementara *rollout* dan setkan fail *config* berdasarkan *volume* yang digunakan berdasarakn fail *config map*.
8. Sambung rollout dan tunggu hingga tamat.
9. *Expose* aplikasi.

4.2 eRating UAT Pipeline Execution Result



5.0 Production Instance Pipeline

5.1 eRating Production Pipeline

```
try {
    timeout(time: 20, unit: 'MINUTES') {
    def appName="erating"
    def projectUat="erating-uat"
    def projectProd="erating-prod"
    def projectCicd="erating-cicd"
    def tag="blue"
    def altTag="green"
    def routeSuffix="apps.osdec.gov.my"
    def gitRepo="github.com/shahrizanrajak/erating_onboard.git"
    def version

    node () {
```

```
    stage ('Promote Prod') {
        timeout(time:4, unit:'HOURS') {
            env.VERSION = input message: 'Promote to PROD?', ok: 'Promote!',
                    parameters: [string(name: 'VERSION', defaultValue: '', description:
'VERSION')]
        }
        // tag for stage
        sh "oc tag ${projectUat}/${appName}:uat-${env.VERSION}
${projectProd}/${appName}:prod-${env.VERSION}"

    }
    stage ('Preparing Deploy') {
        timeout(time:4, unit:'HOURS') {
            input message: "Deploy to Prod?", ok: "Deploy"
        }
        sh "oc project ${projectProd}"
        sh "oc export cm -n ${projectUat} | oc create -n ${projectProd} -f - || true"
        sh "oc get is ${appName} -n ${projectProd} -o jsonpath='{.status.tags[*].tag}' --loglevel=4
> imagetag"
        imagetag = readFile('imagetag').trim()
        echo "find image tage ${imagetag}"
        if (!imagetag.contains(tag)){
            sh "oc tag ${projectProd}/${appName}:prod-${env.VERSION}
${projectProd}/${appName}:${tag}"
            sh "oc new-app ${appName}:${tag} --name=${appName}-${tag} -n ${projectProd} "
            sh "oc rollout cancel dc/${appName}-${tag} -n ${projectProd} || true"
            sh "oc rollout pause dc/${appName}-${tag} -n ${projectProd}"
            sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=htaccess -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-root/src/.htaccess --sub-
path=.htaccess"
            sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=database-
config -t configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/database.php --sub-path=database.php"
            sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/config.php --sub-path=config.php"
            sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=htaccess -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-root/src/.htaccess --sub-
path=.htaccess"
            sh "oc rollout resume dc/${appName}-${tag} -n ${projectProd}"
            sh "oc rollout status dc/${appName}-${tag} -w -n ${projectProd}"
            sh "oc expose svc/${appName}-${tag} --hostname=${appName}-${tag}.${routeSuffix} -n
${projectProd}"
            sh "oc expose svc/${appName}-${tag} --hostname=${appName}.${routeSuffix} --
name=${appName} -n ${projectProd}"
            sh "oc set -n ${projectProd} route-backends ${appName} ${appName}-${tag}=100 --
loglevel=4"
        }
        if (!imagetag.contains(altTag)){
            sh "oc tag ${projectProd}/${appName}:prod-${env.VERSION}
${projectProd}/${appName}:${altTag}"
            sh "oc new-app ${appName}:${altTag} --name=${appName}-${altTag} -n
${projectProd}"
            sh "oc rollout cancel dc/${appName}-${altTag} -n ${projectProd} || true"
            sh "oc rollout pause dc/${appName}-${altTag} -n ${projectProd}"
```

```
            sh "oc set volume dc/${appName}-${altTag} -n ${projectProd} --add --name=htaccess -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-root/src/.htaccess --sub-
path=.htaccess"
            sh "oc set volume dc/${appName}-${altTag} -n ${projectProd} --add --name=database-
config -t configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/database.php --sub-path=database.php"
            sh "oc set volume dc/${appName}-${altTag} -n ${projectProd} --add --name=config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
root/src/application/config/config.php --sub-path=config.php"
            sh "oc set volume dc/${appName}-${altTag} -n ${projectProd} --add --name=htaccess -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-root/src/.htaccess --sub-
path=.htaccess"
            sh "oc rollout resume dc/${appName}-${altTag} -n ${projectProd}"
            sh "oc rollout status dc/${appName}-${altTag} -w -n ${projectProd}"
            sh "oc expose svc/${appName}-${altTag}  --hostname=${appName}-
${altTag}.${routeSuffix} -n ${projectProd}"
            sh "oc delete route ${appName} -n ${projectProd} || true"
            sh "oc expose svc/${appName}-${tag} --hostname=${appName}.${routeSuffix} --
name=${appName} -n ${projectProd}"
            sh "oc patch route ${appName} -p
'{\"spec\":{\"alternateBackends\":[{\"name\":\"${appName}-${altTag}\",\"weight\":0}]}}'"
        }
    }
    stage('Initialize') {
        sh "oc get route ${appName} -n ${projectProd} -o jsonpath='{ .spec.to.name }' --loglevel=4
> activeservice"
        activeService = readFile('activeservice').trim()
        if (activeService == "${appName}-blue") {
        tag = "green"
        altTag = "blue"
        }
        sh "oc get route ${appName}-${tag} -n ${projectProd} -o jsonpath='{ .spec.host }' --
loglevel=4 > routehost"
        routeHost = readFile('routehost').trim()
    }
    stage('Deploy Beta'){
        echo "Deploy Staging tag ${tag}"
        //deploy
        sh "oc rollout pause dc/${appName}-${tag} -n ${projectProd}"
        sh "oc tag ${projectProd}/${appName}:prod-${env.VERSION}
${projectProd}/${appName}:${tag}"
        sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=config -t
configmap --configmap-name=erating-${tag}-cm --mount-path=/opt/app-
root/src/application/config/config.php --sub-path=config.php --overwrite"
        sh "oc rollout resume dc/${appName}-${tag} -n ${projectProd}"
        sh "oc rollout status dc/${appName}-${tag} -w -n ${projectProd}"
    }
    stage("Test Beta") {
        timeout(time:4, unit:'HOURS') {
        input message: "Test deployment: http://${routeHost}. Approve?", id: "approval"
        }
    }
    stage("Go Live") {
        sh "oc rollout pause dc/${appName}-${tag} -n ${projectProd}"
        sh "oc set volume dc/${appName}-${tag} -n ${projectProd} --add --name=config -t
configmap --configmap-name=erating-cm --mount-path=/opt/app-
```

```
root/src/application/config/config.php --sub-path=config.php --overwrite"
          sh "oc rollout resume dc/${appName}-${tag} -n ${projectProd}"
          sh "oc rollout status dc/${appName}-${tag} -w -n ${projectProd}"
          sh "oc set -n ${projectProd} route-backends ${appName} ${appName}-${tag}=100
${appName}-${altTag}=0 --loglevel=4"
       }
   }
}
} catch (err) {
   echo "in catch block"
   echo "Caught: ${err}"
   currentBuild.result = 'FAILURE'
   throw err
}
```

Proses *pipeline* diatas adalah:

1. Minta pengguna masukkan versi aplikasi
2. Salin imej dari *Instance* UAT ke *Instance* PORD berdasarkan versi yang telah dimasukkan.
3. Export fail *config map* dari UAT ke PROD.
4. Periksa *Stream* Imej dengan taq hijau wujud atau tidak.
5. Jika *Stream* imej dengan tag tersebut wujud terus ke langkah 11.
6. Cipta imej dengan tag berdasarkan setiap tag.
7. Cipta aplikasi baru berdasarkan setiap tag.
8. Hentikan *rollout* dan setkan fail config berdasarkan *volume* menggunakan *config map* pada setiap tag.
9. Sambung *rollout* dan tunggu hingga tamat setiap tag.
10. Explose aplikasi untuk setiap tag.
11. Tag ditentukan dengan menggunakan *route suffix*.
12. Periksa servis yang aktif samada biru atau hijau.
13. *Deploy* aplikasi beta pada servis yang tidak aktif
14. Input pengguna untuk ujian beta untuk *deployment* lulus dan disetkan 4 jam menunggu.
15. *Deploy* ke *Production* dan ubah servis yang aktif.

5.2 eRating Production Pipeline Execution Result